(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開2000－29788
(P2000－29788A)
(43)公開日　平成12年１月28日(2000.1.28)

(51)Int.Cl.'	識別記号	ＦＩ		テーマコード(参考)
Ｇ０６Ｆ　12/08		Ｇ０６Ｆ　12/08	Ｍ	５Ｂ００５
			Ｂ	
			Ｕ	

審査請求　有　　請求項の数12　ＯＬ　（全 13 頁）

(21)出願番号　特願平10－199396

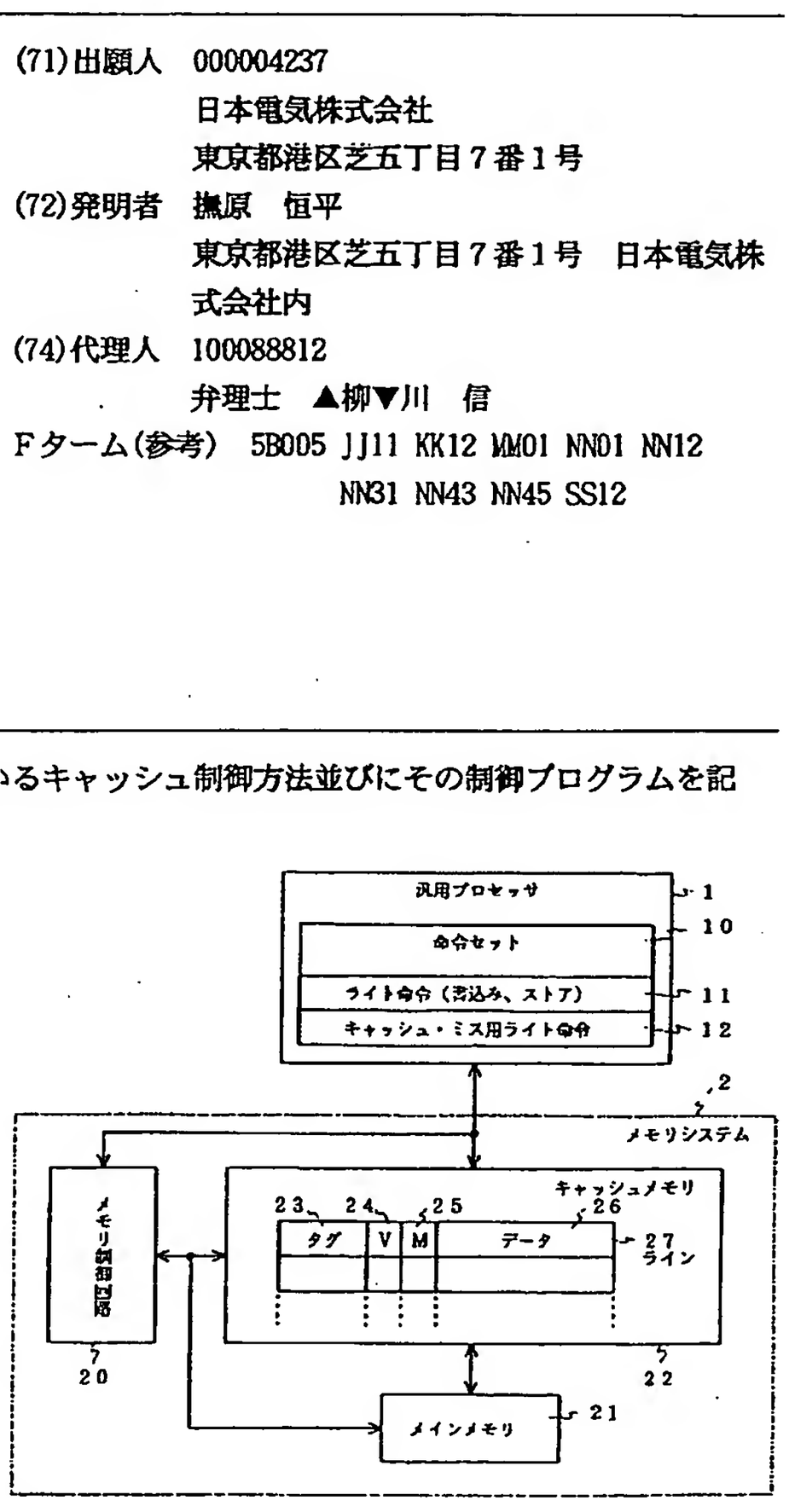
(22)出願日　平成10年７月15日(1998.7.15)

(71)出願人　000004237
日本電気株式会社
東京都港区芝五丁目７番１号
(72)発明者　撫原　恒平
東京都港区芝五丁目７番１号　日本電気株式会社内
(74)代理人　100088812
弁理士　▲柳▼川　信

Ｆターム(参考)　5B005 JJ11 KK12 MM01 NN01 NN12
NN31 NN43 NN45 SS12

(54)【発明の名称】　キャッシュメモリシステム及びそれに用いるキャッシュ制御方法並びにその制御プログラムを記録した記録媒体

(57)【要約】
【課題】　連続したアドレスへ大量のデータを書込む時のキャッシュ・ミス処理を高速化し、そのキャッシュ・ミス処理を含むソフトウェアの実行速度が改善可能なキャッシュメモリシステムを提供する。
【解決手段】　汎用プロセッサ１の命令セット１０中に従来のキャッシュ制御手法に従う通常のライト命令（書込み、ストア）１１と、本発明の一実施例によるキャッシュ制御手法に従うキャッシュ・ミス用ライト命令１２とが夫々別の命令として用意されている。キャッシュ・ミスの処理を行う必要がある場合、キャッシュ・ミス用ライト命令１２は書込みアドレスに相当するメインメモリ２１のアドレスからデータを読込むことなく、キャッシュ・ライン２７を書込みデータで更新し、そのライン２７のＶビット２４及びＭビット２５の双方を“１”にセットする。



【特許請求の範囲】

【請求項１】　メインメモリと、前記メインメモリに格納されたデータの一部を保持するキャッシュメモリとを含むキャッシュメモリシステムであって、前記キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納する前記キャッシュメモリのアドレスを選択する選択手段と、前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを判定する判定手段と、前記判定手段が当該データを無効と判定した時に当該データに対応する前記メインメモリのデータを前記キャッシュメモリに読込むことなく前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込む書込み手段とを有することを特徴とするキャッシュメモリシステム。

【請求項２】　前記選択手段は、データのライト要求時にそのライト対象のアドレスが前記キャッシュメモリのアドレスと一致しない場合及びそのライト対象のアドレスが前記キャッシュメモリのアドレスと一致しても当該アドレスのデータが無効である場合のいずれか一方の場合に前記ライト対象のデータを格納するアドレスを選択するよう構成したことを特徴とする請求項１記載のキャッシュメモリシステム。

【請求項３】　前記判定手段は、前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを示すバリッドビットが無効を示す場合と前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータが前記メインメモリ上のデータよりも新しいか否かを示すモデファイビットが新しくないことを示す場合とのうちのいずれかの場合に前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータを無効と判定するよう構成したことを特徴とする請求項１または請求項２記載のキャッシュメモリシステム。

【請求項４】　前記書込み手段が前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込んだ後に当該アドレスに対応する前記バリッドビットに当該データが有効であることを示す値をセットしかつ前記選択されたアドレスに対応する前記モデファイビットに当該データが書換えられたことを示す値をセットする手段を含むことを特徴とする請求項３記載のキャッシュメモリシステム。

【請求項５】　メインメモリと、前記メインメモリに格納されたデータの一部を保持するキャッシュメモリとを含むキャッシュメモリシステムのキャッシュ制御方法であって、前記キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納する前記キャッシュメモリのアドレスを選択するステップと、その選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを判定するステップと、当該データが無効と判定された時に当該データに対応する前記メインメモリのデータを前記キャッシュメモリに読込むことなく前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込むステップとを有することを特徴とするキャッシュ制御方法。

【請求項６】　前記キャッシュメモリのアドレスを選択するステップは、データのライト要求時にそのライト対象のアドレスが前記キャッシュメモリのアドレスと一致しない場合及びそのライト対象のアドレスが前記キャッシュメモリのアドレスと一致しても当該アドレスのデータが無効である場合のいずれか一方の場合に前記ライトキャッシュ時のデータを格納するアドレスを選択するようにしたことを特徴とする請求項５記載のキャッシュ制御方法。

【請求項７】　前記前記キャッシュメモリのデータが有効か無効かを判定するステップは、前記選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを示すバリッドビットが無効を示す場合と前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータが前記メインメモリ上のデータよりも新しいか否かを示すモデファイビットが新しくないことを示す場合とのうちのいずれかの場合に前記選択されたアドレスに対応する前記キャッシュメモリのデータを無効と判定するようにしたことを特徴とする請求項５または請求項６記載のキャッシュ制御方法。

【請求項８】　前記ライトキャッシュ時のデータが前記選択手段で選択されたアドレスに書込まれた後に当該アドレスに対応する前記バリッドビットに当該データが有効であることを示す値をセットしかつ前記選択されたアドレスに対応する前記モデファイビットに当該データが書換えられたことを示す値をセットするステップを含むことを特徴とする請求項７記載のキャッシュ制御方法。

【請求項９】　コンピュータに、メインメモリに格納されたデータの一部を保持するキャッシュメモリを制御させるためのキャッシュ制御プログラムを記録した記録媒体であって、前記キャッシュ制御プログラムは前記コンピュータに、前記キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納する前記キャッシュメモリのアドレスを選択させ、その選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを判定させ、当該データが無効と判定された時に当該データに対応する前記メインメモリのデータを前記キャッシュメモリに読込むことなく前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込ませることを特徴とするキャッシュ制御プログラムを記録した記録媒体。

【請求項１０】　前記キャッシュ制御プログラムは前記コンピュータに、前記キャッシュメモリのアドレスを選択させる際に、データのライト要求時にそのライト対象のアドレスが前記キャッシュメモリのアドレスと一致しない場合及びそのライト対象のアドレスが前記キャッシ

ュメモリのアドレスと一致しても当該アドレスのデータが無効である場合のいずれか一方の場合に前記ライト対象のデータを格納するアドレスを選択させることを特徴とする請求項９記載のキャッシュ制御プログラムを記録した記録媒体。

【請求項１１】　前記キャッシュ制御プログラムは前記コンピュータに、前記前記キャッシュメモリのデータが有効か無効かを判定させる際に、前記選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを示すバリッドビットが無効を示す場合と前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータが前記メインメモリ上のデータよりも新しいか否かを示すモデファイビットが新しくないことを示す場合とのうちのいずれかの場合に前記選択されたアドレスに対応する前記キャッシュメモリのデータを無効と判定させることを特徴とする請求項９または請求項１０記載のキャッシュ制御プログラムを記録した記録媒体。

【請求項１２】　前記キャッシュ制御プログラムは前記コンピュータに、前記ライトキャッシュ時のデータが前記選択手段で選択されたアドレスに書込まれた後に当該アドレスに対応する前記バリッドビットに当該データが有効であることを示す値をセットさせかつ前記選択されたアドレスに対応する前記モデファイビットに当該データが書換えられたことを示す値をセットさせることを特徴とする請求項１１記載のキャッシュ制御プログラムを記録した記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明はキャッシュメモリシステム及びそれに用いるキャッシュ制御方法並びにその制御プログラムを記録した記録媒体に関し、特にマイクロプロセッサ等のキャッシュメモリシステムにおけるライト動作を高速化するためのキャッシュ制御方法に関する。

【０００２】

【従来の技術】現在、マルチメディア処理はテレビゲーム機やインタネット端末といった安価な民生用家電製品から、カーナビゲーションシステムやパーソナルコンピュータといった比較的高性能な製品まで、様々な情報機器で必要とされている。これらの機器で用いられるマルチメディア処理には音声、オーディオ、ビデオ信号各々の圧縮や伸張、及び２次元や３次元のグラフィックス処理等が含まれる。これらのマルチメディア処理では膨大なデータに対し、多くの演算を行う必要がある。

【０００３】従来、上記のマルチメディア処理においては、機器全体の制御やユーザインタフェースを行う汎用プロセッサに加え、特定のマルチメディア処理を専用に行うＬＳＩ（大規模集積回路）を機器へ組込むことで、処理能力の要求を満たしてきている。

【０００４】しかしながら、最近では、汎用プロセッサの性能向上によって、専用ＬＳＩを組込まなくても、汎用プロセッサ上のソフトウェアによって充分な速度でマルチメディア処理が行えるようになりつつある。

【０００５】専用ＬＳＩを汎用プロセッサ上のマルチメディア処理ソフトウェアで代替すると、専用ＬＳＩという付加ハードウェアが不要になるため、機器を安価に構成することができ、その機能をソフトウェアの変更によって容易に実現することができる。例えば、ソフトウェアの変更によって、オーディオの伸張とビデオ信号の伸張とを切替えることができる。

【０００６】上記のように、ソフトウェアによるマルチメディア処理は利点が多いため、現在の汎用プロセッサでは処理能力が不足する応用でもソフトウェア処理でカバーしようと、汎用プロセッサにマルチメディア処理専用の演算命令を導入し、マルチメディア処理性能を向上させることも一般的である。

【０００７】上記のようなマルチメディア処理命令については、“ＶＩＳ　ＳｐｅｅｄｓＮｅｗ　Ｍｅｄｉａ　Ｐｒｏｃｅｓｓｉｎｇ、”（Ｍａｒｃ　Ｔｒｅｍｂｌａｙほか，ＩＥＥＥ　Ｍｉｃｒｏ，ｐｐ．１０～２０，Ａｕｇ．１９９６）、“Ｉｎｔｅｌ’ｓ　ＭＭＸ　Ｓｐｅｅｄｓ　Ｍｕｌｔｉｍｅｄｉａ”（ＬｉｎｌｅｙＧｗｅｎｎａｐ，Ｍｉｃｒｏｐｒｏｃｅｓｓｏｒ　Ｒｅｐｏｒｔ，Ｖｏｌ．１０，Ｎｏ．３，ｐｐ．１，６～１０，Ｍａｒ．５，１９９６）、“ＮＥＣ　Ｖ８３０Ｒ／ＡＶ　Ｈａｎｄｌｅｓ　Ｒｅａｌ－Ｔｉｍｅ　ＭＰＥＧ－２”（ＪｉｍＴｕｒｌｅｙ，Ｍｉｃｒｏｐｒｏｃｅｓｓｏｒ　Ｒｅｐｏｒｔ，Ｖｏｌ．１１，Ｎｏ．１２，ｐｐ．５，Ｓｅｐ．１５，１９９７）等に記載されている。

【０００８】上記の論文に記載されたマルチメディア処理専用の命令セットはレジスタを分割して複数のデータを格納し、１命令で並列に演算を行うことで、データ並列性の高い処理における処理能力を向上させている。例えば、６４ビット長のレジスタに１６ビット長のデータ４個を格納し、１命令でこれらのデータに対して４並列演算を行える。このようなデータ並列を利用する演算命令を以下、ＳＩＭＤ（Ｓｉｎｇｌｅ－Ｉｎｓｔｒｕｃｔｉｏｎ　Ｍｕｌｔｐｌｅ－Ｄａｔａ）命令と呼ぶ。

【０００９】汎用プロセッサで効率良くマルチメディア処理を行うには、ＳＩＭＤ命令を導入して演算能力を強化すると同時に、メモリシステムを強化し、演算能力に見合ったレートでメモリシステム上のデータが演算器へ供給されるように、また演算後のデータがメモリシステムに格納されるようにすることが重要である。

【００１０】図４～図６を参照して汎用プロセッサ５を用いたシステムの典型的なメモリシステム２の構成及びデータアクセス時の動作について説明する。図４に示すように、汎用プロセッサ５に接続されるメモリシステム２は大容量なメインメモリ２１と、高速なキャッシュメモリ２２とを組合わせて構成され、メインメモリ２１及

びキャッシュメモリ２２を制御するメモリ制御回路２０を備えている。

【００１１】メインメモリ２１に格納されたデータのうち、頻繁にアクセスされる一部のデータのコピーがキャッシュメモリ２２に格納され、汎用プロセッサ５がデータアクセスを行う時に、キャッシュメモリ２２上に目的のデータが存在すれば、低速なメインメモリ２１ではなく、高速なキャッシュメモリ２２をアクセスするよう制御し、プログラムの実行速度を改善する。

【００１２】キャッシュメモリ２２は通常、数百～数千個のライン２７に分割され、ライン２７毎に制御情報を付加し、状態を管理する。１個のライン２７が保持するデータのサイズは通常、１６～６４バイトである。制御情報としてはlライン２７毎に、保持しているデータ２６のメインメモリ２１上のアドレスの上位ビットを示すタグ２３と、データ２６が有効であることを示すバリッド（Ｖ）ビット２４と、データ２６が更新されたことを示すモディファイ（Ｍ）ビット２５とが付加される。

【００１３】Ｖビット２４はそのライン２７のデータ２６が有効か無効かを示す。以下、Ｖビット２４が“１”の時にはそのライン２７のデータ２６を有効とし、“０”の時にはデータ２６を無効とする。

【００１４】Ｍビット２５はそのライン２７のデータ２６が更新されており、キャッシュメモリ２２上のデータ２６がメインメモリ２１上のデータより新しいか、更新されておらず、キャッシュメモリ２２上のデータ２６とメインメモリ２１上のデータとが等しいかを示す。以下、Ｍビットが“１”の時にはキャッシュメモリ２２上のデータ２６がメインメモリ２１上のデータより新しいとし、“０”の時にはキャッシュメモリ２２上のデータ２６とメインメモリ２１上のデータとが等しいものとする。

【００１５】図４に示すように、汎用プロセッサ５とキャッシュメモリ２２とは論理的に独立した機能を持っているが、物理的には同一のＬＳＩチップ上に集積されている場合、別チップだが同一のマルチチップモジュールやカートリッジ内に組込まれている場合等がある。

【００１６】また、図４に示す例では汎用プロセッサ５とメインメモリ２１との間に存在するキャッシュメモリ２２は１段だが、より高性能なメモリシステム２を構成するためにはキャッシュメモリ２２を多段にすることもある。

【００１７】データリード時のキャッシュメモリ２２の制御について、図５を参照しながら説明する。汎用プロセッサ５からキャッシュメモリ２２にアドレスＡｒのデータのリード要求があったとする。メモリ制御回路２０はアドレスＡｒの上位ビットと同じタグ２３の値を持つライン２７を検索する（図５ステップＳ１１）。

【００１８】アドレスＡｒの上位ビットと同じタグ２３の値を持つライン２７が存在し、このラインをラインｉとすると、メモリ制御回路２０はラインｉのＶビット２４を調べ（図５ステップＳ１２）、Ｖビット２４が“１”ならば、ラインｉが目的のデータを含んでいるため（キャッシュ・ヒット）、ラインｉのデータ２６のうち必要とされる部分を汎用プロセッサ５に返す（図５ステップＳ１３）。

【００１９】キャッシュメモリ２２中に、アドレスＡｒの上位ビットと同じタグ２３の値を持つラインが存在しない時、または存在してもＶビット２４が“０”の時、キャッシュメモリ２２中に目的のデータが存在しない（キャッシュ・ミス）ので、メモリ制御回路２０は適当なライン（ラインｊとする）を選択し（図５ステップＳ１４）、メインメモリ２１上のアドレスＡｒのデータをラインｊにコピーする（図５ステップＳ１８）。

【００２０】但し、メモリ制御回路２０は選択したラインｊのＶビット２４及びＭビット２５を調べ（図５ステップＳ１５，Ｓ１６）、それらがともに“１”の時にはラインｊがメインメモリ２１よりも新しいデータを保持しているので、ラインｊのデータをメインメモリ２１の対応するアドレスへ書き戻した後（図５ステップＳ１７）、アドレスＡｒのデータをラインｊにコピーする（図５ステップＳ１８）。

【００２１】その後、メモリ制御回路２０はＶビット２４を“１”に、Ｍビット２５を“０”に夫々設定し（図５ステップＳ１９）、ラインｊがメインメモリ２１のデータの有効なコピーを保持していることを示し、ラインｊのデータのうち必要な部分を汎用プロセッサ５へ返す（図５ステップＳ２０）。このとき、ライン２７の値ｊは通常、アドレスＡｒの値や各ラインの使用状況によって適切に選ばれるが、その手法については公知であるので、その詳細な説明は省略する。

【００２２】次に、データライト時のキャッシュメモリ２２の制御について、図６を参照しながら説明する。ここでは、特にライト・バック方式と呼ばれるキャッシュ制御方式について説明する。尚、このデータライトは汎用プロセッサ５内の命令セット５０に含まれるライト命令（書込み、ストア）５１によって実行される。

【００２３】汎用プロセッサ５からキャッシュメモリ２２にアドレスＡｗのデータのライト要求があったとする。メモリ制御回路２０はアドレスＡｗの上位ビットと同じタグ２３の値を持つライン２７を検索する（図６ステップＳ３１）。

【００２４】アドレスＡｗの上位ビットと同じタグ２３の値を持つライン２７がキャッシュメモリ２２上に存在し、そのラインをラインｉとすると、メモリ制御回路２０はラインｉのＶビット２４を調べ（図６ステップＳ３２）、Ｖビット２４が“１”ならば、ラインｉが目的のデータを含んでいるため（キャッシュ・ヒット）、ラインｉ上のデータを書換える（図６ステップＳ３３）。さらに、メモリ制御回路２０はラインｉのＭビットを

“１”にし（図６ステップＳ３４）、メインメモリ２１上のデータよりキャッシュ・ラインｉ上のデータが新しいことを示す。

【００２５】キャッシュメモリ２２中に、アドレスＡｗの上位ビットと同じタグ値を持つラインが存在しない時、または存在してもＶビット２４が“０”の時、キャッシュメモリ２２中に目的のデータは存在しない（キャッシュ・ミス）ので、メモリ制御回路２０は適当なライン（ラインｊとする）を選択し（図６ステップＳ３５）、メインメモリ２１上のアドレスＡｗのデータをラインｊにコピーする（図６ステップＳ３９）。

【００２６】但し、メモリ制御回路２０は選択したラインｊのＶビット24及びＭビット２５を調べ（図６ステップＳ３６，Ｓ３７）、それらがともに“１”の時にはラインｊがメインメモリ２１よりも新しいデータを保持しているので、ラインｊのデータをメインメモリ２１の対応するアドレスへ書戻した後（図６ステップＳ３８）、アドレスＡｗのデータをラインｊにコピーする（図６ステップＳ３９）。

【００２７】その後、メモリ制御回路２０はラインｊのデータを更新し（図６ステップＳ４０）、ラインｊのＶビット２４及びＭビット２５をともに“１”にし（図６ステップＳ４１）、キャッシュメモリ２２のラインｊ上に有効なデータがあり、メインメモリ２１上のデータよりキャッシュメモリ２２のラインｊ上のデータが新しいことを示す。リードキャッシュ・ミスの時と同様に、ライン２７の値ｊは通常、アドレスＡｗの値や各ラインの使用状況によって適切に選ばれるが、その手法については公知であるので、その詳細な説明は省略する。

【００２８】ライトキャッシュ・ミスの時、アドレスＡｗにある値をいったんキャッシュメモリ２２に読込むのは、一度に書換えられるデータがキャッシュ・ラインｊの一部のみだからである。キャッシュの１ラインは通常、１６～６４バイトのデータを保持するが、１個のデータ書込み命令は通常、１～８バイトのデータを書込む。キャッシュ・ラインｊのデータのうち、データ書込みによって更新されなかったデータは、メインメモリ２１上で対応するデータの値と一致していなくては整合性が保たれない。

【００２９】

【発明が解決しようとする課題】一般的に、マルチメディア処理では大量の演算結果を連続したアドレスへ書込むことが多い。例えば、ビデオ画像の圧縮規格であるＩＴＵ－Ｔ勧告Ｈ．２６１，Ｈ．２６２，Ｈ．２６３，ＭＰＥＧ（Ｍｏｖｉｎｇ　Ｐｉｃｔｕｒｅ　Ｅｘｐｅｒｔｓ　Ｇｒｏｕｐ）－１ビデオ（ＩＳＯ／ＩＥＣ１１１７２，”Ｉｎｆｏｒｍａｔｉｏｎ　Ｔｅｃｈｎｏｌｏｇｙ－Ｃｏｄｉｎｇ　ｏｆ　Ｍｏｖｉｎｇ　Ｐｉｃｔｕｒｅｓ　ａｎｄ　Ａｓｓｏｃｉａｔｅｄ　Ａｕｄｉｏ　ｆｏｒ　Ｄｉｇｉｔａｌ　Ｓｔｏｒａｇｅ　Ｍｅｄｉａ　ｕｐ　ｔｏ１．５Ｍｂｉｔ／ｓ”），ＭＰＥＧ－２ビデオ（ＩＳＯ／ＩＥＣ１３８１８－２，”Ｉｎｆｏｒｍａｔｉｏｎ　Ｔｅｃｈｎｏｌｏｇｙ－Ｇｅｎｅｒｉｃ　Ｃｏｄｉｎｇ　ｏｆ　ｏｆ　Ｍｏｖｉｎｇ　Ｐｉｃｔｕｒｅｓ　ａｎｄ　Ａｓｓｏｃｉａｔｅｄ　Ａｕｄｉｏ”）に従って圧縮されたビットストリームを伸張し、ビデオ画像を再生する場合、再生画像はメインメモリ上に確保したフレームバッファに書込まれる。

【００３０】例えば、ＭＰＥＧ－２　Ｍａｉｎ　Ｌｅｖｅｌで一般的に用いられる解像度である７２０×４８０画素の画像を伸張した場合、１ピクチャあたり輝度信号に３４５，６００バイトの連続した領域を、色差信号２枚に各８６，４００バイトの連続した領域をメインメモリ上に確保する。

【００３１】すなわち、１ピクチャあたり５０６Ｋバイトのメモリ量が必要である。ＭＰＥＧ－２で用いられる双方向動き予測を実現するには最低でも３枚のピクチャを保持する必要があるため、ＭＰＥＧ－２ビデオ伸張ソフトウェアの作業領域は最低でも１．５Ｍバイト必要になる。キャッシュメモリの容量は通常数十～数百Ｋバイトであり、これだけの量のデータをすべて保持することはできない。したがって、フレームバッファアクセスではデータキャッシュ・ミスが頻発し、高速なＭＰＥＧ－２ビデオ伸張ソフトウェア作成のための障害となっている。

【００３２】さらに、１秒間に３０枚の画像をリアルタイム再生するには、毎秒１４．８Ｍバイトのデータをメインメモリに書込む必要がある。この書込みがデータキャッシュ・ミスを引き起こしていると考えると、キャッシュメモリとメインメモリとの整合性を保つために、伸張データの書込みの前に、メインメモリからキャッシュメモリへ同じ量のデータを読込んでいるため、この２倍、毎秒２９．６Ｍバイトのデータアクセスがメインメモリに対して発生していることになる。この量は、例えば、３２ｂｉｔ幅６６ＭＨｚ動作のバスのバンド幅の１１．５％にも達し、その処理性能上無視することができない。

【００３３】図７に連続したアドレスへデータをライトした時のキャッシュメモリとメインメモリとの間のデータのやり取りと、キャッシュメモリ及びメインメモリ上のデータの変化とを示す。

【００３４】典型的な例として、キャッシュの１ラインのデータサイズを１６バイト、ライトするデータのサイズを４バイトと仮定する。ここで、メインメモリのアドレスＡｗから４個の４バイト・データＮ０，Ｎ１，Ｎ２，Ｎ３を１個ずつライトした場合を考える。

【００３５】さらに、キャッシュメモリ上にはアドレスＡｗのデータが最初存在しないものとし、アドレスＡｗのデータを保持するキャッシュ・ラインｉは不定値Ｕ０，Ｕ１，Ｕ２，Ｕ３を格納しているものとする。

【００３６】最初に、アドレスＡｗに値Ｎ０をライトした場合の動作を考える。アドレスＡｗへの値Ｎ０のライト要求が発生した時［図７の（１１）参照］、アドレスＡｗのデータはキャッシュメモリ上に存在しないためにキャッシュ・ミスが発生し、メインメモリ上のアドレスＡｗから１ライン１６バイト分のデータＭ０，Ｍ１，Ｍ２，Ｍ３［図７の４７参照］がキャッシュ・ラインｉに読込まれる［図７の（１７）及び４２参照］。その後、ラインｉのデータのうちアドレスＡｗからの４バイト分のデータに相当するＭ０がＮ０によって上書きされる［図７の（１２）及び４３参照］。

【００３７】次に、隣接するアドレスＡｗ＋４に４バイトのデータＮ１がライトされる［図７の（１３）参照］。アドレスＡｗから１ライン１６バイト分のデータはキャッシュメモリ上に存在するため、このライトはキャッシュにヒットし、ラインｉ上のアドレスＡｗ＋４に相当するデータをＮ１に更新する［図７の４４参照］。

【００３８】三番目に、隣接するアドレスＡｗ＋８に、４バイトのデータＮ２がライトされる［図７の（１４）参照］。アドレスＡｗから１ライン１６バイト分のデータはキャッシュメモリ上に存在するため、このライトはキャッシュにヒットし、ラインｉ上のアドレスＡｗ＋８に相当するデータをＮ２に更新する［図７の４５参照］。

【００３９】最後に、隣接するアドレスＡｗ＋１２に、４バイトのデータＮ３がライトされる［図７の（１５）参照］。アドレスＡｗから１ライン１６バイト分のデータはキャッシュメモリ上に存在するため、このライトはキャッシュにヒットし、ラインｉ上のアドレスＡｗ＋１２に相当するデータをＮ３に更新する［図７の４６参照］。

【００４０】以上、Ｎ０，Ｎ１，Ｎ２，Ｎ３のライトによって、キャッシュ・ラインｉ上のデータはすべて書換えられており、メインメモリ上のアドレスＡｗから１ライン１６バイト分のデータ４７よりも新しい。したがって、ラインｉに別のアドレスのデータを格納する時には、まずラインｉのデータをメインメモリに書戻す必要がある［図７の（１６）及び４８参照］。

【００４１】そこで、本発明の目的は上記の問題点を解消し、連続したアドレスへ大量のデータを書込む時のキャッシュ・ミス処理を高速化することができ、そのキャッシュ・ミス処理を含むソフトウェアの実行速度を改善することができるキャッシュメモリシステム及びそれに用いるキャッシュ制御方法並びにその制御プログラムを記録した記録媒体を提供することにある。

【００４２】

【課題を解決するための手段】本発明によるキャッシュメモリシステムは、メインメモリと、前記メインメモリに格納されたデータの一部を保持するキャッシュメモリとを含むキャッシュメモリシステムであって、前記キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納する前記キャッシュメモリのアドレスを選択する選択手段と、前記選択手段で選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを判定する判定手段と、前記判定手段が当該データを無効と判定した時に当該データに対応する前記メインメモリのデータを前記キャッシュメモリに読込むことなく前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込む書込み手段とを備えている。

【００４３】本発明によるキャッシュ制御方法は、メインメモリと、前記メインメモリに格納されたデータの一部を保持するキャッシュメモリとを含むキャッシュメモリシステムのキャッシュ制御方法であって、前記キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納する前記キャッシュメモリのアドレスを選択するステップと、その選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを判定するステップと、当該データが無効と判定された時に当該データに対応する前記メインメモリのデータを前記キャッシュメモリに読込むことなく前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込むステップとを備えている。

【００４４】本発明によるキャッシュ制御プログラムを記録した記録媒体は、コンピュータに、メインメモリに格納されたデータの一部を保持するキャッシュメモリを制御させるためのキャッシュ制御プログラムを記録した記録媒体であって、前記キャッシュ制御プログラムは前記コンピュータに、前記キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納する前記キャッシュメモリのアドレスを選択させ、その選択されたアドレスに対応する前記キャッシュメモリのデータが有効か無効かを判定させ、当該データが無効と判定された時に当該データに対応する前記メインメモリのデータを前記キャッシュメモリに読込むことなく前記ライトキャッシュ時のデータを前記選択手段で選択されたアドレスに書込ませている。

【００４５】すなわち、本発明のキャッシュメモリシステムは、データと、保持しているデータのメインメモリ上のアドレスの上位ビットを示すタグと、データが有効であることを示すバリッド（Ｖ）ビットと、データが更新されたことを示すモディファイ（Ｍ）ビットとを含む複数のキャッシュ・ラインから構成されている。

【００４６】本発明のキャッシュメモリシステムは、キャッシュメモリへの書込み時、キャッシュ中に書込みアドレスの上位ビットと一致するタグを持ち、かつＶビットが“１”であるラインが存在せず、キャッシュ・ミスの処理を行う必要がある時、書込みアドレスに相当するメインメモリのアドレスからデータを読込むことなく、キャッシュ・ラインを書込みデータで更新し、Ｖビット

及びMビットの双方を“１”にセットしている。

【００４７】このため、メインメモリのアドレスからデータを読込む時間を削減できる。よって、連続したアドレスへ大量のデータを書込む時のキャッシュ・ミス処理を高速化することが可能となり、そのキャッシュ・ミス処理を含むソフトウェアの実行速度を改善することが可能となる。

【００４８】

【発明の実施の形態】次に、本発明の一実施例について図面を参照して説明する。図１は本発明の一実施例によるキャッシュメモリシステムの構成を示すブロック図である。図において、本発明の一実施例によるキャッシュメモリシステムは汎用プロセッサ１と、汎用プロセッサ１に接続されるメモリ・システム２とから構成されている。

【００４９】汎用プロセッサ１はライト命令（書込み、ストア）１１とキャッシュミス用ライト命令１２とを含む命令セット１０を持ち、ソフトウェア実行時、メモリシステム２に格納されたデータをリード・ライトする。尚、命令セット１０はＲＯＭ（リードオンリメモリ）やＩＣ（集積回路）メモリ等に格納可能である。

【００５０】メモリシステム２は大容量なメインメモリ２１と、高速なキャッシュメモリ２２とを組合わせて構成され、メインメモリ２１及びキャッシュメモリ２２を制御するメモリ制御回路２０を備えている。

【００５１】キャッシュメモリ２２は数百～数千個のライン２７に分割され、ライン２７毎に制御情報を付加し、状態を管理する。１個のライン２７が保持するデータのサイズは１６～６４バイトである。

【００５２】制御情報としては１ライン２７毎に、保持しているデータ２６のメインメモリ２１上のアドレスの上位ビットを示すタグ２３と、データ２６が有効であることを示すバリッド（Ｖ）ビット２４と、データ２６が更新されたことを示すモディファイ（Ｍ）ビット２５とが付加される。

【００５３】Ｖビット２４はそのライン２７のデータ２６が有効か無効かを示す。以下、Ｖビット２４が“１”の時にはそのライン２７のデータ２６を有効とし、“０”の時にはデータ２６を無効とする。

【００５４】Ｍビット２５はそのライン２７のデータ２６が更新されており、キャッシュメモリ２２上のデータ２６がメインメモリ２１上のデータより新しいか、更新されておらず、キャッシュメモリ２２上のデータ２６とメインメモリ２１上のデータとが等しいかを示す。以下、Ｍビットが“１”の時にはキャッシュメモリ２２上のデータ２６がメインメモリ２１上のデータより新しいとし、“０”の時にはキャッシュメモリ２２上のデータ２６とメインメモリ２１上のデータとが等しいものとする。

【００５５】本発明の一実施例においても、データリード時の動作は図５に示す従来の技術と同様である。すなわち、汎用プロセッサ１からキャッシュメモリ２２にアドレスＡｒのデータのリード要求があったとする。メモリ制御回路２０はアドレスＡｒの上位ビットと同じタグ２３の値を持つライン２７を検索する（図５ステップＳ１１）。

【００５６】アドレスＡｒの上位ビットと同じタグ２３の値を持つライン２７が存在し、このラインをラインｉとすると、メモリ制御回路２０はラインｉのＶビット２４を調べ（図５ステップＳ１２）、Ｖビット２４が“１”ならば、ラインｉが目的のデータを含んでいるため（キャッシュ・ヒット）、ラインｉのデータ２６のうち必要とされる部分を汎用プロセッサ１に返す（図５ステップＳ１３）。

【００５７】キャッシュメモリ２２中に、アドレスＡｒの上位ビットと同じタグ２３の値を持つラインが存在しない時、または存在してもＶビット２４が“０”の時、キャッシュメモリ２２中に目的のデータが存在しない（キャッシュ・ミス）ので、メモリ制御回路２０は適当なライン（ラインｊとする）を選択し（図５ステップＳ１４）、メインメモリ２１上のアドレスＡｒのデータをラインｊにコピーする（図５ステップＳ１８）。

【００５８】但し、メモリ制御回路２０は選択したラインｊのＶビット２４及びＭビット２５を調べ（図５ステップＳ１５，Ｓ１６）、それらがともに“１”の時にはラインｊがメインメモリ２１よりも新しいデータを保持しているので、ラインｊのデータをメインメモリ２１の対応するアドレスへ書き戻した後（図５ステップＳ１７）、アドレスＡｒのデータをラインｊにコピーする（図５ステップＳ１８）。

【００５９】その後、メモリ制御回路２０はＶビット２４を“１”に、Ｍビット２５を“０”に夫々設定し（図５ステップＳ１９）、ラインｊがメインメモリ２１のデータの有効なコピーを保持していることを示し、ラインｊのデータのうち必要な部分を汎用プロセッサ１へ返す（図５ステップＳ２０）。このとき、ライン２７の値ｊは通常、アドレスＡｒの値や各ラインの使用状況によって適切に選ばれるが、その手法については公知であるので、その詳細な説明は省略する。

【００６０】図２は本発明の一実施例によるデータライト時の動作を示すフローチャートである。これら図１及び図２を参照して本発明の一実施例によるデータライト時の動作について説明する。

【００６１】汎用プロセッサ１からキャッシュメモリ２２にアドレスＡｗのデータのライト要求があったとする。メモリ制御回路２０はアドレスＡｗの上位ビットと同じタグ２３の値を持つライン２７を検索する（図２ステップＳ１）。

【００６２】アドレスＡｗの上位ビットと同じタグ２３の値を持つライン２７が存在し、このラインをラインｉ

とすると、メモリ制御回路２０はラインｉのＶビット２４を調べ（図２ステップＳ２）、Ｖビット２４が“１”ならば、ラインｉが目的のデータを含んでいるため（キャッシュ・ヒット）、ラインｉ上のデータを書換える（図２ステップＳ３）。さらに、メモリ制御回路２０はラインｉのＭビットを“１”にし（図２ステップＳ４）、メインメモリ２１上のデータよりキャッシュ・ラインｉ上のデータが新しいことを示す。

【００６３】キャッシュメモリ２２中に、アドレスＡｗの上位ビットと同じタグ２３の値を持つラインが存在しない時、または存在してもＶビット２４が“０”の時、キャッシュメモリ２２中に目的のデータは存在しない（キャッシュ・ミス）ので、メモリ制御回路２０は適当なライン（ラインｊとする）を選択し（図２ステップＳ５）、その後、ラインｊにデータをライトする（図２ステップＳ９）。

【００６４】但し、メモリ制御回路２０は選択したラインｊのＶビット２４及びＭビット２５を調べ（図２ステップＳ６，Ｓ７）、それらがともに“１”の時にはラインｊがメインメモリ２１よりも新しいデータを保持しているので、ラインｊのデータをメインメモリ２１の対応するアドレスへ書き戻した後（図２ステップＳ８）、ラインｊにデータをライトする（図２ステップＳ９）。

【００６５】その後、メモリ制御回路２０はラインｊのＶビット２４及びＭビット２５をともに“１”にし（図２ステップＳ１０）、メインメモリ２１上のデータよりキャッシュ・ラインｉ上のデータが新しいことを示す。リードキャッシュ・ミス時と同様に、ライン２７の値ｊは通常、アドレスＡｗの値や各ラインの使用状況によって適切に選ばれるが、その手法については公知であるので、その詳細な説明は省略する。

【００６６】図３は本発明の一実施例によるデータライト時の動作を示す図である。この図３を参照して本発明の一実施例によってキャッシュ・ミス時の処理がどのように高速化されるのかを説明する。

【００６７】図３は図７に示す動作と同様に、連続したアドレスへデータをライトした時のキャッシュメモリ制御手法で制御されるキャッシュメモリ２２とメインメモリ２１との間のデータのやり取りと、キャッシュメモリ２２及びメインメモリ２１上のデータの変化とを示す。

【００６８】この場合、図７と同様に、キャッシュの１ラインのデータサイズを１６バイト、ライトするデータのサイズを４バイトと仮定し、メインメモリ２１のアドレスＡｗから４個の４バイトのデータＮ０，Ｎ１，Ｎ２，Ｎ３を１個ずつライトした場合を考える。さらに、これも図７と同様に、開始時にキャッシュメモリ２２上にはアドレスＡｗのデータが存在しないものとし、アドレスＡｗのデータを保持する予定のキャッシュ・ラインｉは不定値Ｕ０，Ｕ１，Ｕ２，Ｕ３を格納しているものとする［図３の３１参照］。

【００６９】最初に、アドレスＡｗに値Ｎ０をライトする要求が発生した場合［図３の（１）参照］の動作を考える。このとき、アドレスＡｗのデータはキャッシュ上に存在しないため、キャッシュ・ミスが発生するが、本発明の一実施例では従来例とは異なり、メインメモリ２１上のアドレスＡｗから１ライン分のデータＭ０，Ｍ１，Ｍ２，Ｍ３をキャッシュ・ラインｉに読込むことが行われず、ラインｉのデータのうちアドレスＡｗからの４バイト分のデータに相当するＵ０がＮ０によって上書きされるのみである［図３の（２）及び３２参照］。

【００７０】ラインｉのデータのうちアドレスＡｗ＋４、Ａｗ＋８、Ａｗ＋１２に相当するデータは不定値Ｕ１，Ｕ２，Ｕ３であり、メインメモリ２１の値Ｍ１，Ｍ２，Ｍ３を反映しない。しかしながら、キャッシュのヒット・ミス判定時にはラインｉがメインメモリ２１上のアドレスＡｗから１ライン１６バイト分のデータを保持しているとみなされる。

【００７１】次に、隣接するアドレスＡｗ＋４に４バイトのデータＮ１がライトされる［図３の（３）参照］。ヒット・ミス判定時にはアドレスＡｗから１ライン１６バイト分のデータがキャッシュメモリ２２上に存在すると扱われるため、このデータＮ１のライトはキャッシュにヒットし、ラインｉ上のアドレスＡｗ＋４に相当するデータがＮ１に更新される［図３の３３参照］。

【００７２】三番目に、隣接するアドレスＡｗ＋８に４バイトのデータＮ２がライトされる［図３の（４）参照］。ヒット・ミス判定時にはアドレスＡｗから１ライン１６バイト分のデータがキャッシュメモリ２２上に存在すると扱われるため、このデータＮ２のライトはキャッシュにヒットし、ラインｉ上のアドレスＡｗ＋８に相当するデータがＮ２に更新される［図３の３４参照］。

【００７３】最後に、隣接するアドレスＡｗ＋１２に４バイトのデータＮ３がライトされる［図３の（５）参照］。ヒット・ミス判定時にはアドレスＡｗから１ライン１６バイト分のデータがキャッシュメモリ２２上に存在すると扱われるため、このライトはキャッシュにヒットし、ラインｉ上のアドレスＡｗ＋１２に相当するデータがＮ３に更新される［図３の３５参照］。

【００７４】以上、Ｎ０，Ｎ１，Ｎ２，Ｎ３のライトによってキャッシュ・ラインｉ上のデータがすべて書き換えられており、メインメモリ２１上のアドレスＡｗから１ライン１６バイト分のデータ３６よりも新しい。したがって、ラインｉに別のアドレスのデータを格納する時には、まずラインｉのデータをメインメモリ２１に書き戻す必要がある［図３の（６）及び３７参照］。

【００７５】図３に示す本発明の一実施例による動作例と図７に示す従来の動作例とを比較すると、アドレスＡｗに値Ｎ３をライトした後には同一の結果が得られている。すなわち、キャッシュ・ラインｉは値Ｎ０，Ｎ１，Ｎ２，Ｎ３を保持している［図３の３５及び図７の４

６］。

【００７６】しかしながら、従来例ではこの結果を得るためにメインメモリ２１からキャッシュ・ラインｉへのリード［図７の（１７）］を行っているのに対し、本発明の一実施例ではこのリードを行っていない。したがって、本発明の一実施例では従来例よりも高速に結果を得ることができる。

【００７７】図３において、キャッシュ・ミス時にメインメモリ２１からキャッシュ・ラインへのデータリードが不要な理由を説明する。あるキャッシュ・ラインｉ上のデータが複数のライト動作ですべて更新される場合、最初のライトが起こしたキャッシュ・ミスでメインメモリ２１からキャッシュ・ラインｉへデータを転送しても、メインメモリ２１から転送されたデータは複数のライト動作ですべて上書きされてしまうため、キャッシュ・ラインｉが保持する最終結果はライトデータの値のみに依存し、メインメモリ２１の内容には依存しない。したがって、メインメモリ２１からキャッシュ・ラインへのデータリードは不要である。

【００７８】一般的には、あるキャッシュ・ライン上のデータが複数のライト動作ですべて更新される保証はないため、メインメモリ２１の内容と対応するキャッシュ・ラインのデータの内容との整合性を保つにはキャッシュ・ミス時にメインメモリ２１の内容をキャッシュ・ラインにリードする必要がある。しかしながら、マルチメディアアプリケーション等ではしばしばキャッシュ・ライン上のデータが複数のライト動作ですべて更新されることが保証される。

【００７９】例えば、圧縮されたビデオ画像をソフトウェアで伸張することを考える。伸張されたビデオ画像は数百キロバイト程度のフレームバッファに連続して書込まれるため、フレームバッファ領域に対応するキャッシュ・ラインはすべての内容が更新される。マルチメディアアプリケーション、特にビデオ画像を扱うアプリケーションでは大量のデータをフレームバッファに書込むことが多く、不要なリード動作を省略することによる高速化の効果も大きい。

【００８０】図６に示す従来のキャッシュ制御手法に従う通常のライト命令（書込み、ストア）と、図２に示す本発明の一実施例によるキャッシュ制御手法に従うライト命令とは汎用プロセッサ１の命令セット１０中に夫々別の命令として用意し、アプリケーションソフトウェアのプログラマが使い分ける。

【００８１】なぜなら、あるキャッシュ・ライン上のデータが複数のライト動作ですべて更新されるかどうかを汎用プロセッサ１上の何らかのハードウェア機構で判断することは不可能だからである。また、高級言語コンパイラが同様の判断を下すことも困難だからである。

【００８２】このように、汎用プロセッサ１の命令セット１０中に従来のキャッシュ制御手法に従う通常のライト命令（書込み、ストア）１１と、図２に示す本発明の一実施例によるキャッシュ制御手法に従うキャッシュ・ミス用ライト命令１２とを夫々別の命令として用意し、キャッシュ・ミスの処理を行う必要がある時にキャッシュ・ミス用ライト命令１２によって書込みアドレスに相当するメインメモリ２１のアドレスからデータを読込むことなく、キャッシュ・ラインを書込みデータで更新し、そのラインのＶビット及びＭビットの双方を“１”にセットすることによって、あるキャッシュ・ライン上のデータをすべて更新する場合、ライト・ミス時のメインメモリ２１からキャッシュ・ラインへのリードを省略してその処理を高速化することができる。

【００８３】このため、メインメモリ２１のアドレスからデータを読込む時間を削減することができる。よって、連続したアドレスへ大量のデータを書込む時のキャッシュ・ミス処理を高速化することができ、そのキャッシュ・ミス処理を含むソフトウェアの実行速度を改善することができる。

【００８４】

【発明の効果】以上説明したように本発明によれば、メインメモリと、メインメモリに格納されたデータの一部を保持するキャッシュメモリとを含むキャッシュメモリシステムにおいて、キャッシュメモリに対するライトキャッシュ・ミスが発生した際に当該ライトキャッシュのデータを格納するキャッシュメモリのアドレスを選択し、その選択されたアドレスに対応するキャッシュメモリのデータが有効か無効かを判定し、当該データが無効と判定された時に当該データに対応するメインメモリのデータをキャッシュメモリに読込むことなくライトキャッシュ時のデータを選択されたアドレスに書込むことによって、連続したアドレスへ大量のデータを書込む時のキャッシュ・ミス処理を高速化することができ、そのキャッシュ・ミス処理を含むソフトウェアの実行速度を改善することができるという効果がある。

【図面の簡単な説明】

【図１】本発明の一実施例によるキャッシュメモリシステムの構成を示すブロック図である。

【図２】本発明の一実施例によるデータライト時の動作を示すフローチャートである。

【図３】本発明の一実施例によるデータライト時の動作を示す図である。

【図４】従来例によるキャッシュメモリシステムの構成を示すブロック図である。

【図５】従来例によるデータリード時の動作を示すフローチャートである。

【図６】従来例によるデータライト時の動作を示すフローチャートである。

【図７】従来例によるデータライト時の動作を示す図である。

【符号の説明】

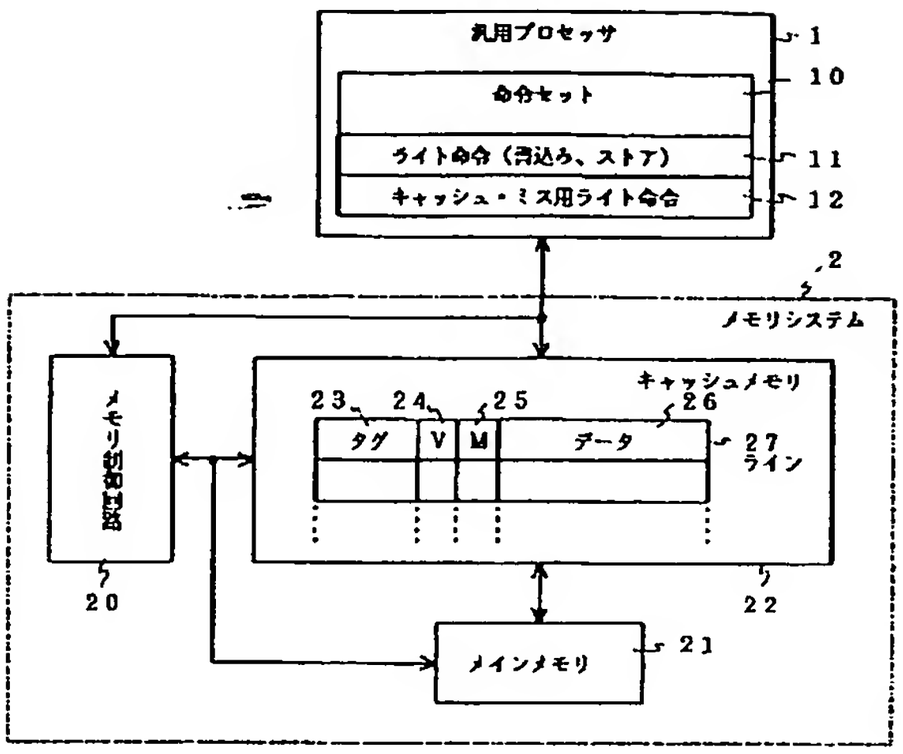
17

１　汎用プロセッサ
２　メモリシステム
１０　命令セット
１１　ライト命令（書込み、ストア）
１２　キャッシュ・ミス用ライト命令
２０　メモリ制御回路
２１　メインメモリ

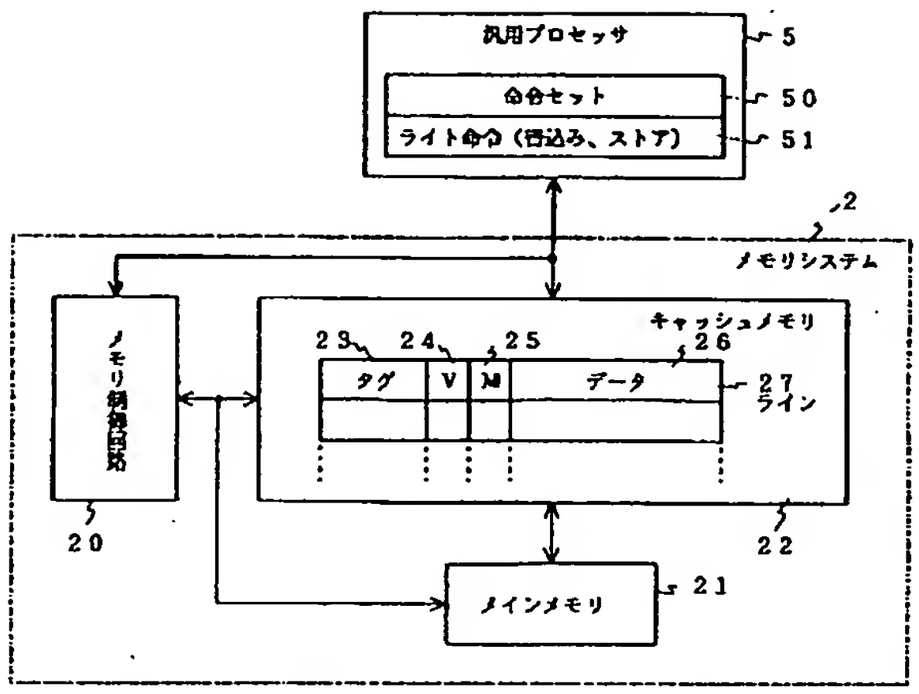
18

＊２２　キャッシュメモリ
２３　タグ
２４　バリッドビット
２５　モディファイビット
２６　データ
２７　キャッシュ・ライン
＊

【図１】

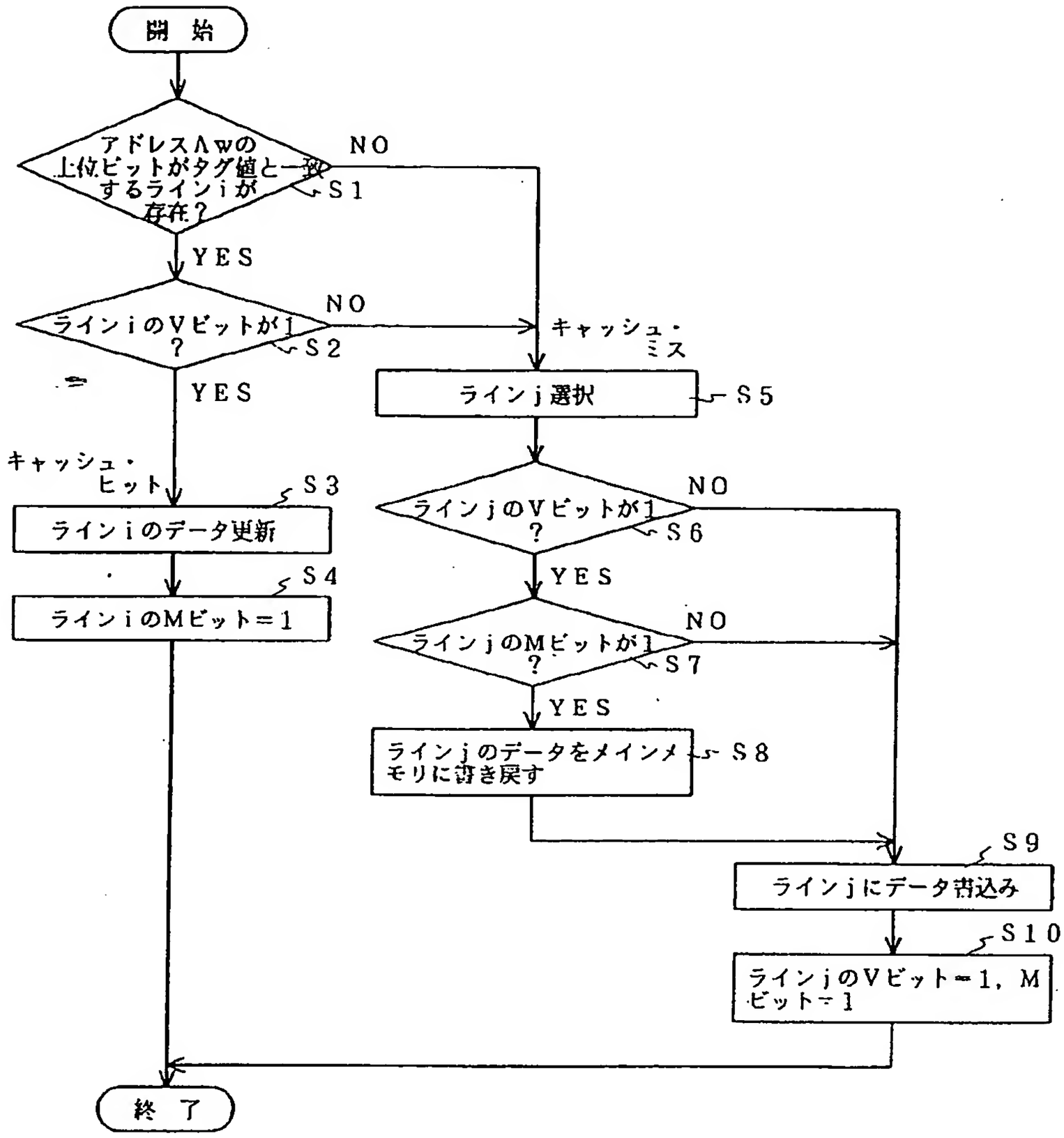


【図４】

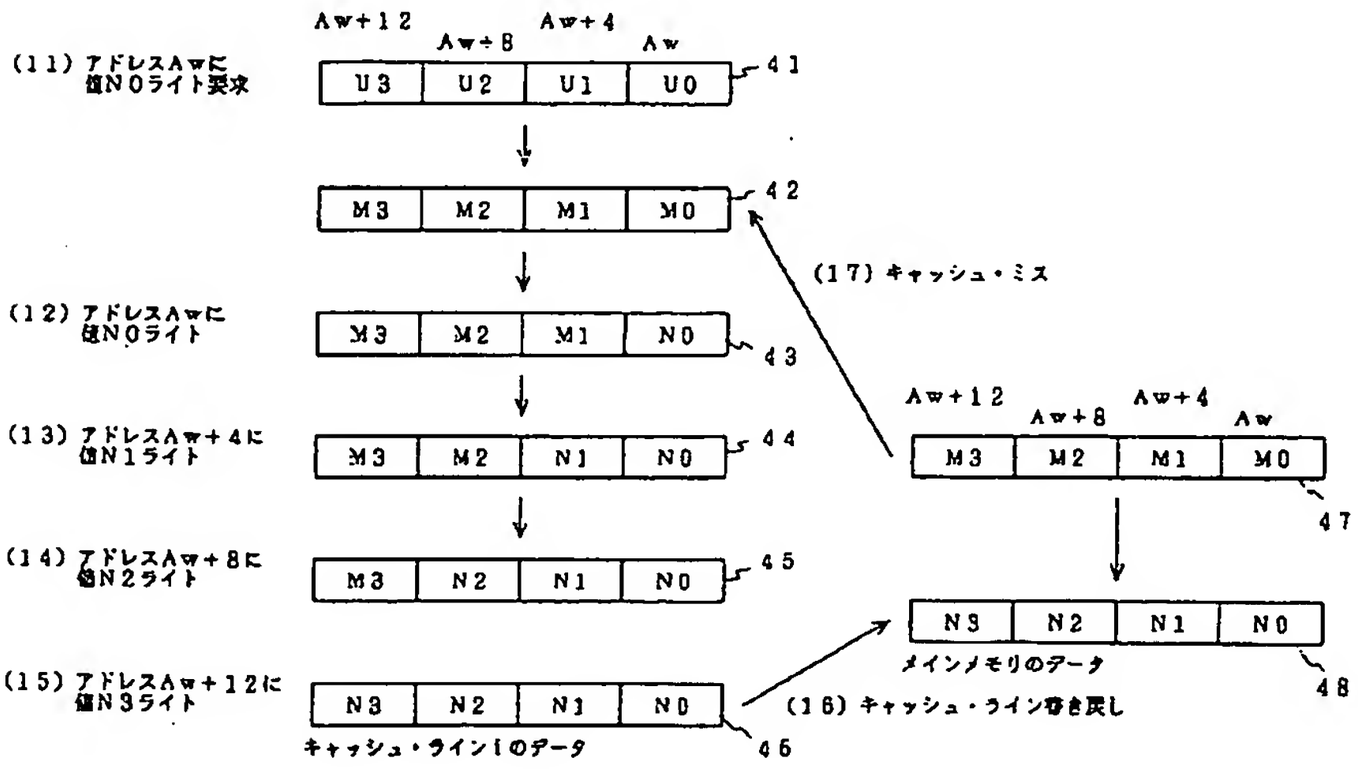


【図３】

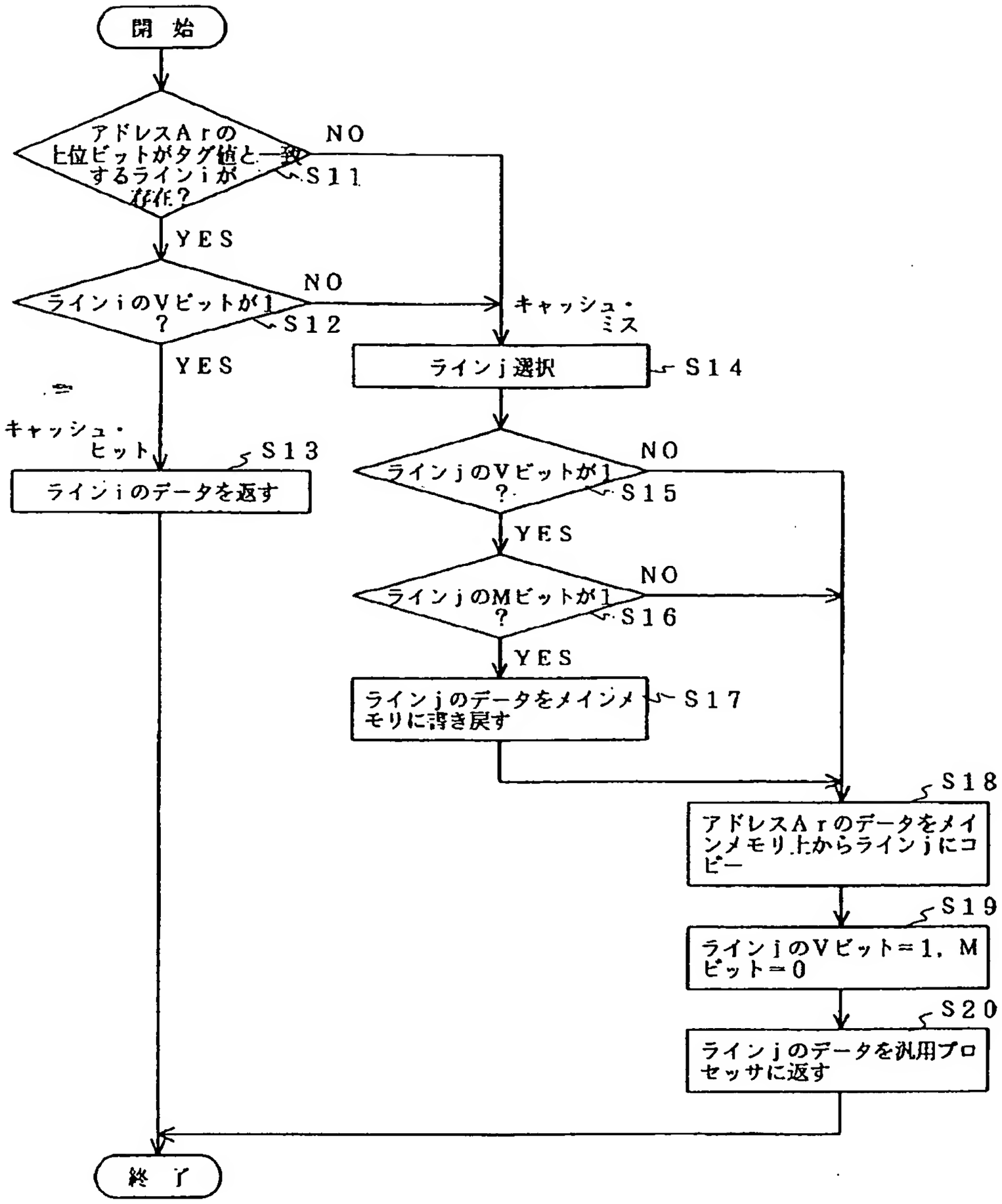
【図２】


開　始
アドレスAwの上位ビットがタグ値と一致するラインiが存在？
NO
S1
YES
ラインiのVビットが1？
NO
S2
キャッシュ・ミス
YES
キャッシュ・ヒット
ラインj選択
S5
S3
ラインiのデータ更新
ラインjのVビットが1？
NO
S6
YES
S4
ラインiのMビット＝1
ラインjのMビットが1？
NO
S7
YES
ラインjのデータをメインメモリに書き戻す
S8
S9
ラインjにデータ書込み
S10
ラインjのVビット＝1，Mビット＝1
終　了

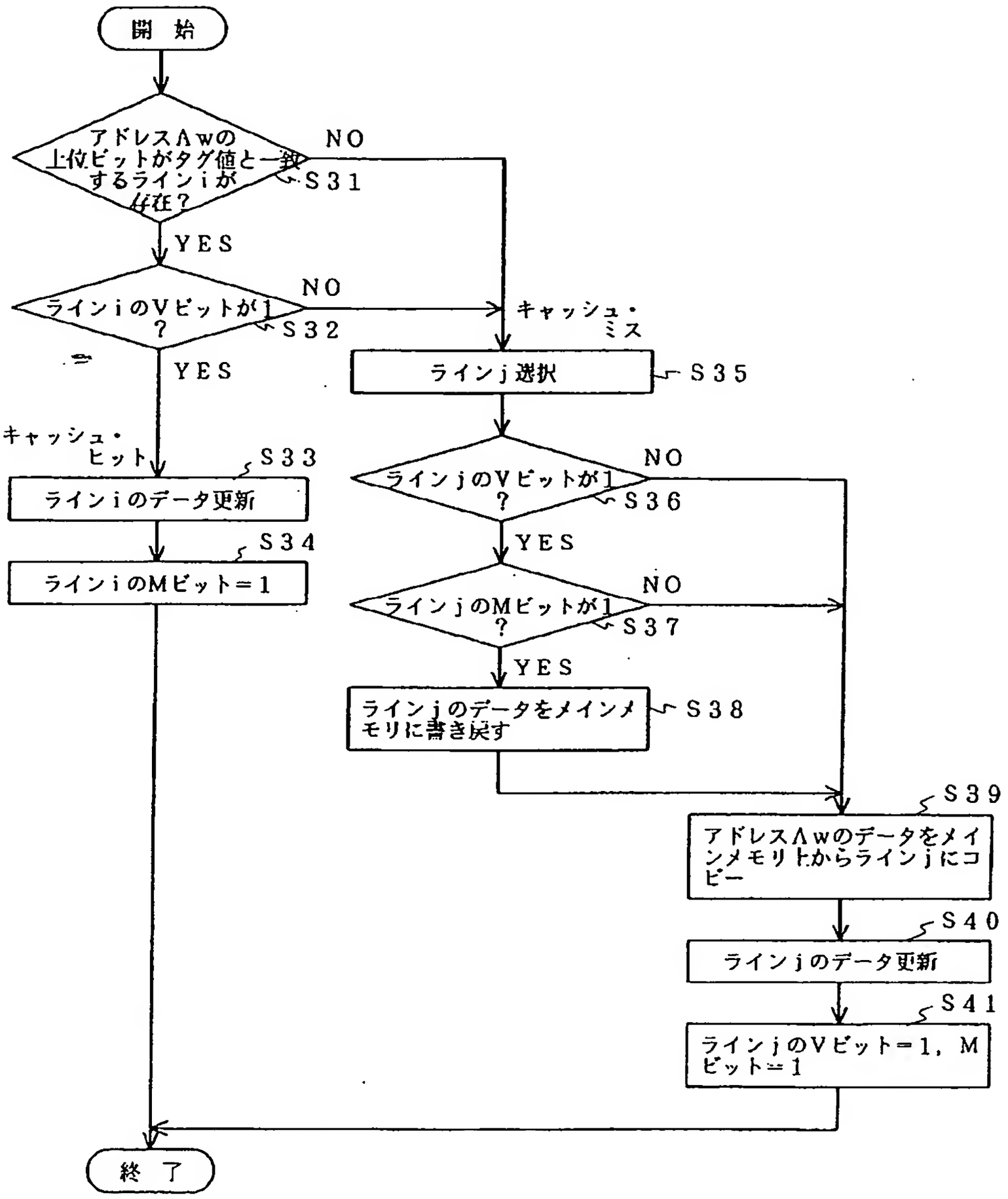
【図７】


Aw+12
Aw+8
Aw+4
Aw
(11) アドレスAwに値N0ライト要求
U3
U2
U1
U0
41
M3
M2
M1
M0
42
(12) アドレスAwに値N0ライト
N0
43
(13) アドレスAw+4に値N1ライト
N1
44
(14) アドレスAw+8に値N2ライト
N2
45
(15) アドレスAw+12に値N3ライト
N3
キャッシュ・ラインiのデータ
46
(17) キャッシュ・ミス
47
メインメモリのデータ
48
(16) キャッシュ・ライン書き戻し

【図５】


開　始
アドレスＡｒの上位ビットがタグ値と一致するラインｉが存在？
NO
Ｓ１１
YES
ラインｉのＶビットが１？
NO
Ｓ１２
キャッシュ・ミス
YES
キャッシュ・ヒット
ラインｊ選択
Ｓ１４
Ｓ１３
ラインｉのデータを返す
ラインｊのＶビットが１？
NO
Ｓ１５
YES
ラインｊのＭビットが１？
NO
Ｓ１６
YES
ラインｊのデータをメインメモリに書き戻す
Ｓ１７
Ｓ１８
アドレスＡｒのデータをメインメモリ上からラインｊにコピー
Ｓ１９
ラインｊのＶビット＝１，Ｍビット＝０
Ｓ２０
ラインｊのデータを汎用プロセッサに返す
終　了

【図６】


開　始
アドレスＡｗの上位ビットがタグ値と一致するラインｉが存在？
NO
S３１
YES
ラインｉのＶビットが１？
NO
S３２
キャッシュ・ミス
YES
キャッシュ・ヒット
S３３
ラインｉのデータ更新
S３４
ラインｉのＭビット＝１
ラインｊ選択
S３５
ラインｊのＶビットが１？
NO
S３６
YES
ラインｊのＭビットが１？
NO
S３７
YES
ラインｊのデータをメインメモリに書き戻す
S３８
S３９
アドレスＡｗのデータをメインメモリ上からラインｊにコピー
S４０
ラインｊのデータ更新
S４１
ラインｊのＶビット＝１，Ｍビット＝１
終　了

[0060]
Figure 2 is a flow chart showing the operation during data writing according to the embodiment of the present invention. The operation during data writing according to the embodiment of the present invention will be described with reference to Figure 1 and Figure 2.

[0061]
Assume that a write request to write the address AW data to the cache memory 22 is provided from the general purpose processor. The memory control circuit 20 searches for a line 27 having a tag 23 value that matches the upper bit of the address Aw (Figure 2, step S1).

[0062]
If a line 27 having a tag 23 value matching the upper bit of the address Aw exists and this line is the line i, the memory control circuit 20 checks the V bit 24 of the line i (Figure 2, step S 2). If the V bit 24 is "1", the data of the line i is rewritten (Figure 2 the step S 3) because the line i includes the target data (cache hit). Also, the memory control circuit 20 sets the M bit of the line i to "1" (Figure 2, step S 4), to indicate that the data on the cache line i is newer than the data in the memory 21.

[0063]
In the cache memory 22, when the line having a tag 23 value that matches the upper bit of the address Aw does not exist or even when the line exists, the V bit 24 is "0", the target data is not present in the cache memory 22 (cache miss). Therefore, the memory control circuit 20 selects an appropriate line (the line j) (Figure 2, step S 5) and then writes the data to the line j (Figure 2, step S 9).

[0064]
In this regard, the memory control circuit 20 checks the V bit 24 and M bit 25 of the selected line j (Figure 2, step S 6 and S 7). If the both are "1", the line j holds data which is newer than the data in the main memory 21. Thus, the data of line j is written back to the corresponding address of the main memory 21 (Figure 2, step S 8), after which, data is written into the line j (Figure 2, step S 9).

[0065]
Subsequently, the memory control circuit 20 sets the V bit 24 and M bit 25 of the line j to "1" (Figure 2, step S 10), to indicate that the data on the cache line i is newer than the data in the main memory 21. In the same manner as in the case of a read cache miss, the value j of the line 27 is appropriately selected according to the address Aw value and the use condition of each line. As this method is publicly known, detailed description shall be omitted.

[0066]
Figure 3 shows the operation during data writing according to the embodiment of the present invention. How to speed up the process during a cache miss will be described based on the embodiment of the present invention with reference to Figure 3.

[0067]
As in the operation shown in Figure 7, Figure 3 shows the exchange of data between the main memory 21 and the cache memory 22 that is controlled by the cache memory control method, when data is written into consecutive addresses. It also shows the changing of data in the cache memory 22 and the main memory 21.

[0068]
In this case, as shown in Figure 7, it is assumed that the data size of a cache line is 16 bytes and the size of data to be written is 4 bytes, and subsequently, these four 4 byte data N0, N1, N2, and N3 are written one by one from the address Aw of the main memory 22. Also, as in Figure 7, the data of address Aw is not present in the cache memory 22 at the start of the process and the cache line i, which is scheduled to hold the address Aw data, stores indefinite values: U0, U1, U2, and U3 (Refer to 31 of Figure 3).

[0069]
First, the operation in the case where a request to write the value N0 into the address Aw is provided is considered (Refer to (1) of Figure 3). As the address Aw data is not present in the cache, a cache miss takes place. However, unlike the conventional technology, in the embodiment of the present invention, data M0, M1, M2 and M3 for one line from the address Aw of the main memory 21 is not read into the cache line i, and among the data in line i, only U0 which corresponds to the 4 byte data from the address Aw of the line i data is overwritten with N0 (Refer to (2) and 32 of Figure 3).

[0070]
Among the line i data, the data corresponding to the addresses Aw+4, Aw+8 and Aw+12 are the indefinite U1, U2 and U3, which do not reflect the values M1, M2 and M3 of the main memory 21. However, in the judgment of a cache hit/miss, line i is regarded to hold 16 bytes of data per line, from the address Aw of the main memory 21.

[0071]
Next, the 4 byte data N1 is written to the adjacent address Aw+4 (Refer to (3) of Figure 3). In the hit/miss judgment, since it is regarded that 16 bytes of data per line from the address Aw is present in the cache memory 22, a hit occurs in the cache for this data N1 write operation, and data corresponding to the address Aw+4 in the line i is updated with N1 (Refer to 33 of Figure 3).

[0072]
Subsequently, the 4 byte data N2 is written to the adjacent address Aw+8 (Refer to (4) of Figure 3). In the hit/miss judgment, since it is regarded that 16 bytes of data per line from the address Aw is present in the cache memory 22, a hit occurs in the cache for this data N2 write operation, and data corresponding to the address Aw+8 in the line i is updated with N2 (Refer to 34 of Figure3).

[0073]
Finally, the 4 byte data N3 is written into the adjacent address Aw+12 (Refer to (5) of Figure 3). In the hit/miss judgment, since it is regarded that 16 bytes of data per line from the address Aw is present in the cache memory 22, a hit occurs in the cache for this data N3 write operation, and data corresponding to the address Aw+12 in the line i is updated with N3 (Refer to 35 of Figure 3).

[0074]
As described above, all data on the cache line i is rewritten with N0, N1, N2 and N3, and the data is newer than the 16-byte line data 36 from the address Aw in the main memory 21. Therefore, when a different address data is stored in the line i, the data in the line i should be written back to the main memory 21 [Refer to 37 and (6) of Figure 3].

[0075]
Comparing the operation from the embodiment of the present invention shown in Figure

3 with the conventional operation example shown in Figure 7, after the value N3 is written into the address Aw, the same result is obtained. In other words, the cache line i holds the values N0, N1, N2 and N3 (Refer to 35 of Figure 3 and 46 of Figure 7).

[0076]
However, whereas reading from the main memory 21 to the cache line i is executed in the conventional example ((17) of Figure 7) in order to obtain this result, such reading is not executed in the embodiment of the present invention. Thus, in the embodiment of the present invention, the result can be obtained faster than in the conventional example.

[0077]
Figure 3 describes the reason why reading of data from the main memory 21 to a cache line is not necessary at the time of a cache miss. In the case where all data in a cache line i is updated by a number of write operations, even when data from the main memory 21 is transferred to the cache line i due to a cache miss caused by the first write operation, all data transferred from the main memory is overwritten by plural write operations. Therefore, the final result stored in the line i is only dependent on the write data value and is not dependent on the content of the main memory 21. Thus, the reading of data from the main memory 21 to the cache line is not necessary.

[0078]
Generally, all data in a cache line is not guaranteed to be updated by plural write operations. Therefore, to maintain the consistency between the content of the main memory 21 and the content of the corresponding cache line data, it is necessary to read the content of the main memory 21 to the cache line when a cache miss occurs. However, in multimedia applications, all data on a cache line is guaranteed to be updated by plural write operations.

[0079]
For example, it is assumed that a compressed video image is decompressed by using software. Since the decompressed video image is successively written into an approximately several hundred-kilobyte frame buffer, all content of the cache line corresponding to the frame buffer area is updated. In multimedia applications, especially in applications for video images, a large amount of data is often written into the frame buffer, contributing to the acceleration of the process by omitting unnecessary read operations.

[0080]
A general write command (write, store) based on the conventional cache control method shown in Figure 6 and the write command based on the cache control method from the embodiment of the present invention shown in Figure 2 are prepared as separate commands in a command set 10 in the general purpose processor 1 and used in accordance with the intended use by an application software programmer.

[0081]
This is because it is impossible to judge if all data on a cache line can be updated by plural write operations in any hardware unit on the general purpose processor 1. In addition, it is difficult for a high level language compiler to make such a judgment.

[0082]
As just described, a general write command (write, store) 11, based on the conventional cache control method in the command set 10 of the general purpose processor 1 and the write command 12 for a cache miss, based on the cache control method in the embodiment of the present invention shown in Figure 2 are prepared as separate commands. When cache miss processing is required, the cache line is updated with the write data and the V bit and M bit of the line is set to "1" without reading data from the address of the main memory 21 corresponding to the write address according to the command 12 for a cache miss, thereby resulting to the updating of all data on the cache line. In this case, it is possible to omit the reading from the main memory 21 to the cache line, and speed up the process.

[0083]
This makes it possible to reduce the time for reading data from the address of the main memory 21. Thus, cache miss processing in writing a large amount of data to consecutive addresses can be speeded up and the execution speed of software including the process for a cache miss can be improved.

[0084]
[Effects of the Invention]
As described above, according to the present invention, in the main memory and the cache memory system including a cache memory for maintaining part of data stored in the main memory, a cache memory address is selected to store the write cache data at

the time of a cache miss. It is judged whether the cache memory data corresponding to the selected address is valid or invalid. If the data is judged as being invalid, data in the write cache is written into the selected address without reading the main memory data corresponding to the data to the cache memory. This makes it possible to speed up cache miss processing in writing a large amount of data to consecutive addresses and improve the execution speed of software including cache miss processing.

Figure 1

【図1】

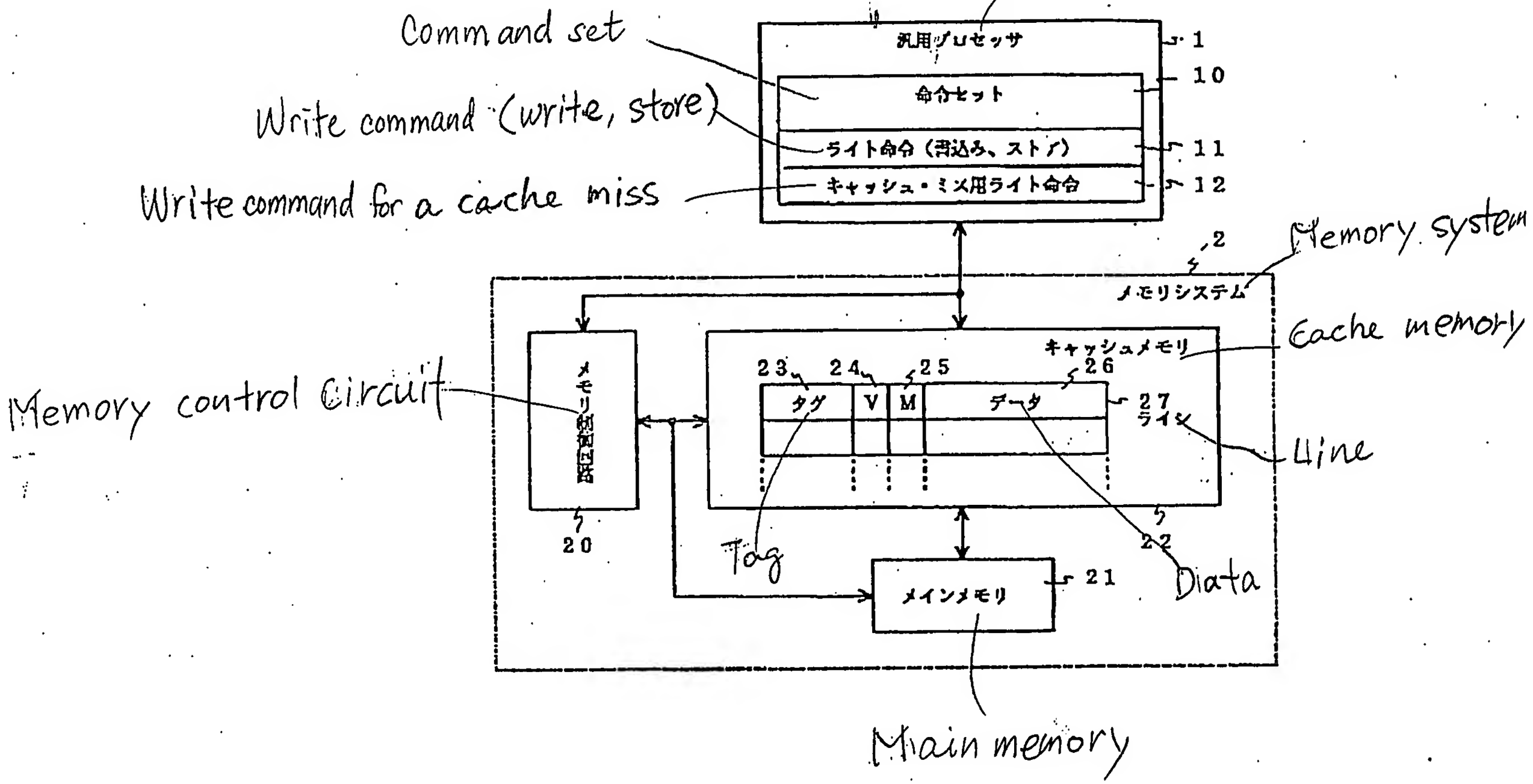

General purpose processor
Command set
Write command (write, store)
Write command for a cache miss
汎用プロセッサ
1
命令セット
10
ライト命令（書込み、ストア）
11
キャッシュ・ミス用ライト命令
12
2
Memory system
メモリシステム
Memory control circuit
Cache memory
キャッシュメモリ
23
24
25
26
タグ
V
M
データ
27
ライン
Line
メモリ制御回路
20
Tag
22
Data
メインメモリ
21
Main memory

Figure 2

【図2】

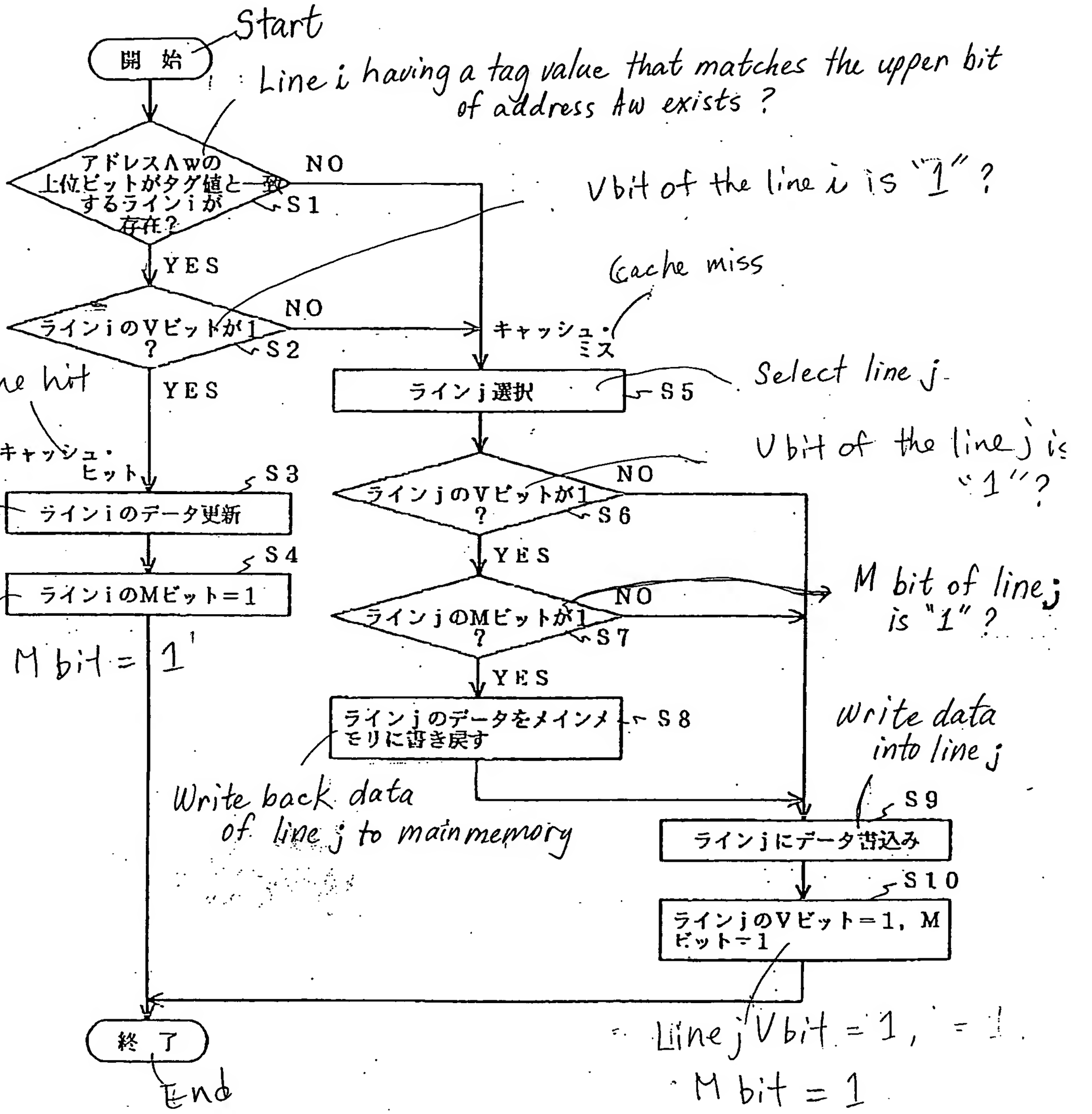
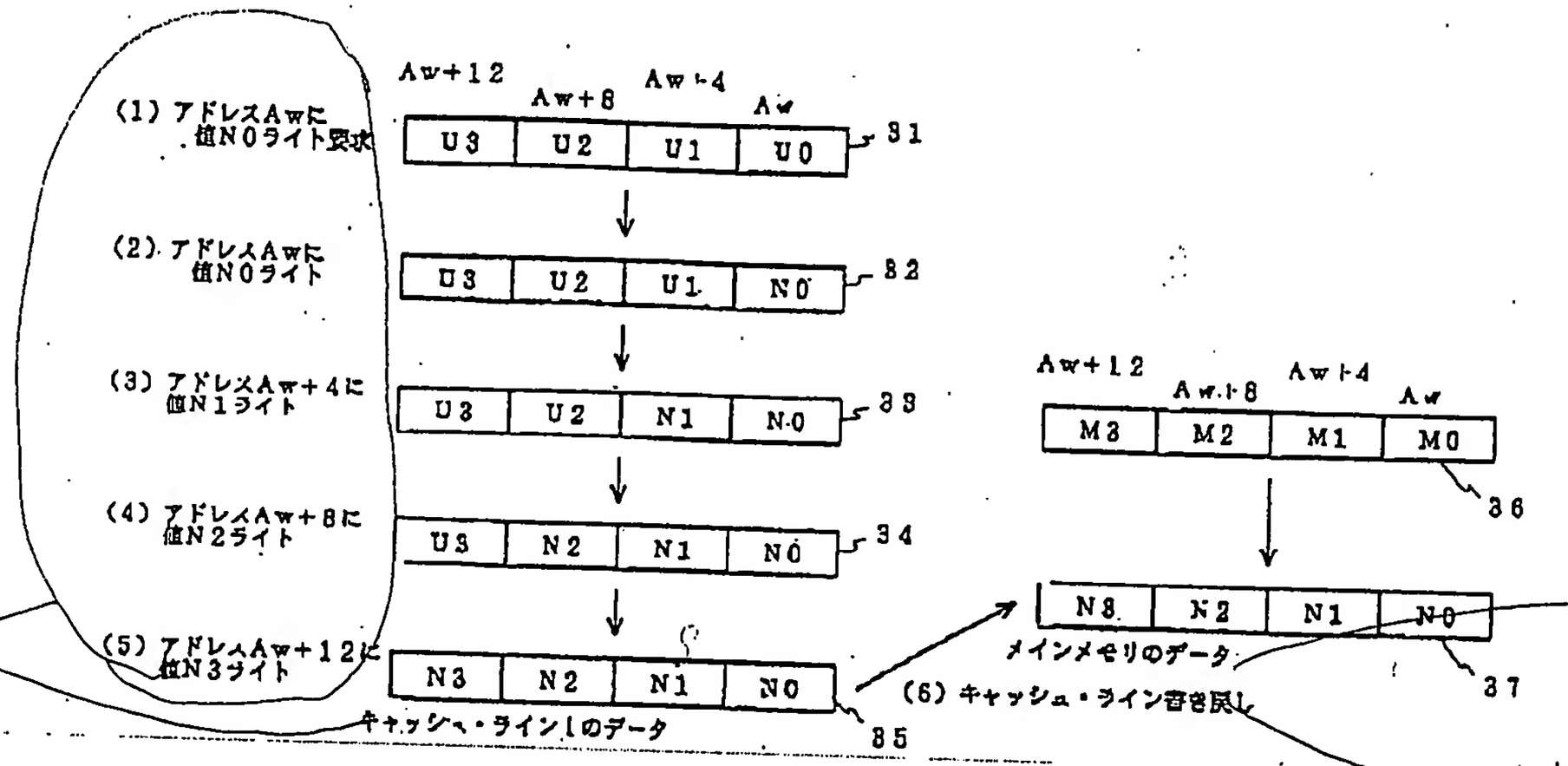

Start
開始
Line i having a tag value that matches the upper bit of address Aw exists ?
アドレスAwの上位ビットがタグ値と一致するラインiが存在?
NO
S1
YES
Vbit of the line i is "1" ?
ラインiのVビットが1?
NO
S2
Cache miss
キャッシュ・ミス
YES
Cache hit
キャッシュ・ヒット
Select line j
ラインj選択
S5
Ubit of the line j is "1" ?
S3
Update data of line i
ラインiのデータ更新
ラインjのVビットが1?
NO
S6
YES
S4
ラインiのMビット=1
Line i M bit = 1
M bit of line j is "1" ?
ラインjのMビットが1?
NO
S7
YES
YES
ラインjのデータをメインメモリに書き戻す
S8
Write back data of line j to main memory
Write data into line j
S9
ラインjにデータ書込み
S10
ラインjのVビット=1, Mビット=1
Line j V bit = 1, M bit = 1
終了
End

Figure 3

【図3】



e line 1 data

Main memory data

Write back cache line

Request for writing a value N0 into address Aw

) Write value N0 into address Aw

) Write value N1 into address Aw+4

Write value N2 into address Aw+8

) Write value N3 into address Aw+12

Figure 6

【図6】

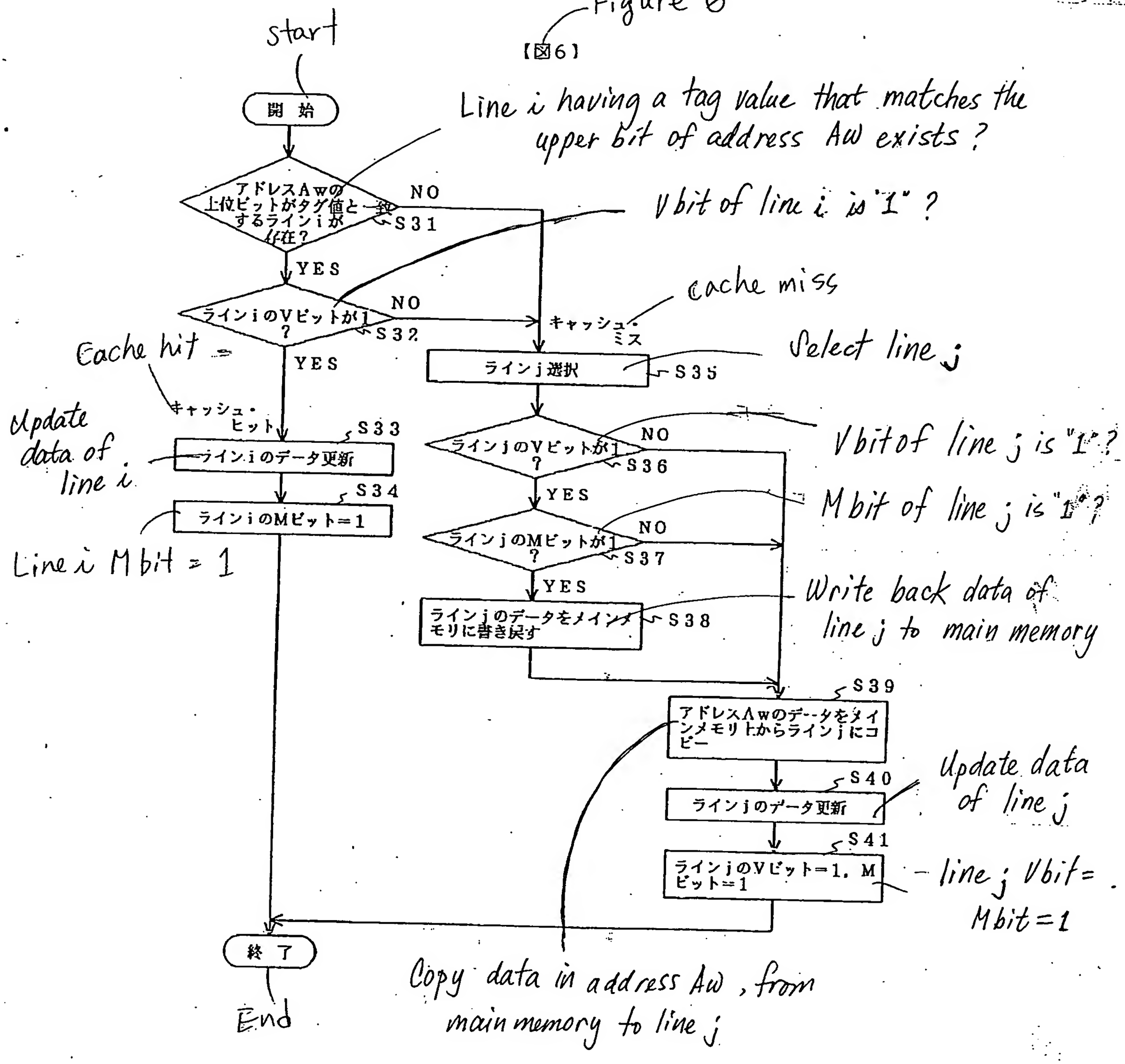
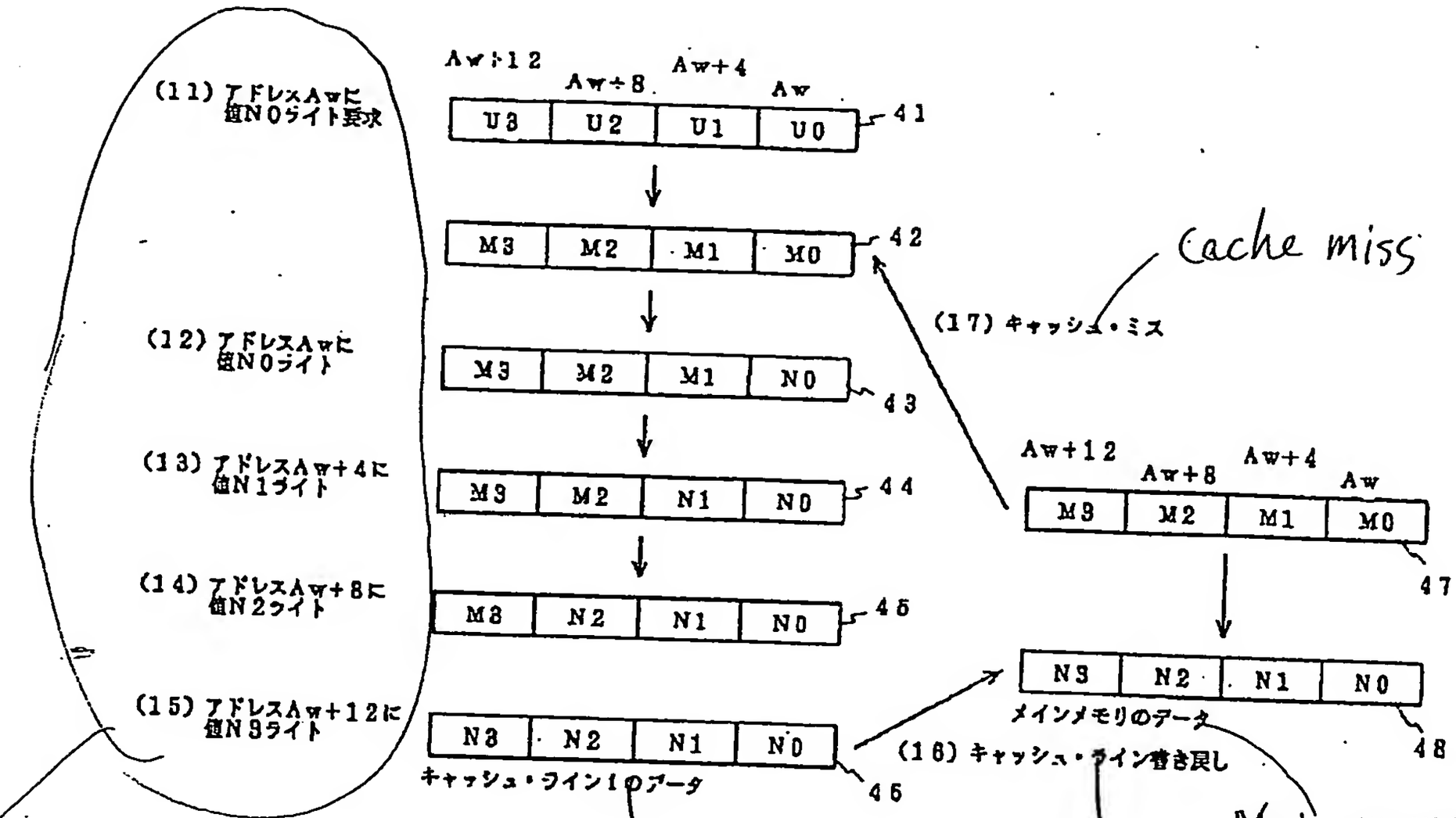

start
開始
アドレスAwの上位ビットがタグ値と一致するラインiが存在?
NO
S31
YES
ラインiのVビットが1?
NO
S32
YES
キャッシュ・ヒット
キャッシュ・ミス
ラインi選択
S35
ラインjのVビット=1?
NO
S36
YES
ラインjのMビット=1?
NO
S37
YES
ラインjのデータをメインメモリに書き戻す
S38
S39
アドレスAwのデータをメインメモリからラインjにコピー
S40
ラインjのデータ更新
S41
ラインjのVビット=1, Mビット=1
S33
ラインiのデータ更新
S34
ラインiのMビット=1
終了
End
Line i having a tag value that matches the upper bit of address Aw exists?
V bit of line i is "1"?
Cache miss
Select line j
V bit of line j is "1"?
M bit of line j is "1"?
Write back data of line j to main memory
Update data of line j
line j V bit = M bit = 1
Cache hit
Update data of line i
Line i M bit = 1
Copy data in address Aw, from main memory to line j
End

Figure 7

【図7】



Cache miss

Data of the cache line 1

Write back

Main memory da

cache line

(11) Request for writing value N0 into address Aw

(12) Write value N0 into address Aw

(13) Write value N1 into address Aw+4

(14) Write value N2 into address Aw+8

(15) Write value N3 into address Aw+12